Learning a Multi-Domain Curriculum for Neural Machine Translation

Wei Wang
Google Research
wangwe@google.com

Ye Tian
Google Research
ytian@google.com

Jiquan Ngiam
Google Brain
jngiam@google.com

Yinfei Yang
Google Research
yinfeiy@google.com

Isaac Caswell
Google Research
icaswell@google.com

Zarana Parekh
Google Research
zarana@google.com

Abstract

Most data selection research in machine translation focuses on improving a single domain. We perform data selection for multiple domains at once. This is achieved by carefully introducing instance-level domain-relevance features and automatically constructing a training curriculum to gradually concentrate on multi-domain relevant and noise-reduced data batches. Both the choice of features and the use of curriculum are crucial for balancing and improving all domains, including out-of-domain. In large-scale experiments, the multi-domain curriculum simultaneously reaches or outperforms the individual performance and brings solid gains over no-curriculum training.

1 Introduction

In machine translation (MT), data selection, e.g., (Moore and Lewis, 2010; Axelrod et al., 2011), has remained as a fundamental and important research topic. It has played a crucial role in domain adaptation by selecting domain-matching training examples, or data cleaning (aka denoising) by selecting high-quality examples. So far, the most extensively studied scenario assumes a single domain to improve.

It becomes both technically challenging and practically appealing to build a large-scale multi-domain neural machine translation (NMT) model that performs simultaneously well on multiple domains at once. This requires addressing research challenges such as catastrophic forgetting (Goodfellow et al., 2014) at scale and data balancing. Such a model can easily find potential use cases, i.e., as a solid general service, for downstream transfer learning, for better deployment efficiency, or for transfer learning across datasets.

Unfortunately, existing single-domain data-selection methods do not work well for multiple domains. For example, improving the translation accuracy of one domain will often hurt that of another (van der Wees et al., 2017; Britz et al., 2017), and improving model generalization across all domains by clean-data selection (Koehn et al., 2018) may not promise optimization of a particular domain. Multiple aspects need to be considered for training a multi-domain model.

		Static	Dynamic
Single	domain	Y	Y
	noise	Y	Y
Multi	domain	Y	N (Our Work)
	noise	N	

Table 1: Data selection and data mixing research in NMT. 'Y': There is previous research that studies this case. 'N': No previous research has studied this case.

This paper presents a dynamic data selection method to multi-domain NMT. Things we do differently from previous work in mixing data are the choice of instance-level features and the employment of a multi-domain curriculum that is additionally able to denoise. These are crucial for mixing and improving all domains, including out-of-domain. We experiment with large datasets at different noise levels and show that the resulting models meet our requirements.

2 Related Work

In MT, research that is most relevant to our work is data selection and data mixing, both being concerned with how to sample examples to train an MT model, usually for domain adaptation. Table 1 categorizes previous research by two aspects and shows where our work stands. These two aspects are:

1. Is the method concerned with a single domain or multiple domains?

2. Does the method use data statically or dynamically?

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7711–7723

Static data selection for a single domain. Moore and Lewis (2010) select in-domain data for n-gram language model (LM) training. It is later generalized by Axelrod et al. (2011) to select parallel data for training MT models. Chen and Huang (2016); Chen et al. (2016) use classifiers to select domain data. Clean-data selection (Koehn et al., 2019, 2018; Junczys-Dowmunt, 2018) reduces harmful data noise to improve translation quality across domains. All these works select a data subset for a single "domain"[1] and treat the selected data as a static/flat distribution.

Dynamic data selection for a single domain. Static selection has two shortcomings: it discards data and it treats all examples equally after selection. When data is scarce, any data could be helpful, even if it is out of domain or noisy[2]. Dynamic data selection is introduced to "sort" data from least in-domain to most in-domain. Training NMT models on data sorted this way effectively takes advantage of transfer learning. Curriculum learning (CL) (Bengio et al., 2009) has been used as a formulation for dynamic data selection. Domain curricula (van der Wees et al., 2017; Zhang et al., 2019) are used for domain adaptation. Model stacking (Sajjad et al., 2017; Freitag and Al-Onaizan, 2016) is a practical idea to build domain models. CL is also used for denoising (Kumar et al., 2019; Wang et al., 2018a,b), and for faster convergence and improved general quality (Zhang et al., 2018; Platanios et al., 2019). Wang et al. (2018a) introduce a curriculum for training efficiency. In addition to data sorting/curriculum, instance/loss weighting (Wang et al., 2017; Chen et al., 2017; Wang et al., 2019b) has been used as an alternative. CL for NMT represents the SOTA data-selection method, but most existing works target at a single "domain", be it a specific domain or the "denoising domain".

Static data mixing for multiple domains. When mixing data from multiple domains, a fundamental challenge is to address catastrophic forgetting (Goodfellow et al., 2014)–training an NMT model to focus on one domain can likely hurt another (van der Wees et al., 2017; Britz et al., 2017). Britz et al. (2017) learn domain-discerning (or -invariant) network representation with a domain discriminator network for NMT. The methods, however, require that domain labels are available in data. Tars and Fishel (2018) cluster data and tag each cluster as multi-domain NMT training data, but the method treats data in each cluster as a flat distribution. Farajian et al. (2017) implement multi-domain NMT by on-the-fly data retrieval and adaptation per sentence, at increased inference cost. Most existing methods (or experiment setups) have the following problems: (i) They mix data statically. (ii) They don't consider the impact of data noise, which is a source of catastrophic forgetting. (iii) Experiments are carried out with small datasets, without separate examination on the data regularization effect. (iv) They do not examine out-of-domain performamce.

Automatic data balancing for multi-domains. (Wang et al., 2020) automatically learn to weight (flat) data streams of multi-languages (or "domains"). We perform dynamic data selection and regularization through a mulit-domain curriculum.

Automatic curriculum learning. Our work falls under automatic curriculum construction (Graves et al., 2017) and is directly inspired by Tsvetkov et al. (2016), who learn to weight and combine instance-level features to form a curriculum for an embedding learning task, through Bayesian Optimization. A similar idea (Ruder and Plank, 2017) is used to improve other NLP tasks. Here, we use the idea for NMT to construct a multi-domain data selection scheme with various selection scores at our disposal. The problem we study is connected to the more general multi-objective optimization problem. Duh (2018) uses Bandit learning to tune hyper-parameters such as the number of network layers for NMT.

More related work. Previously, catastrophic forgetting has mostly been studied in the continued-training setup (Saunders et al., 2019; Khayrallah et al., 2018), to refer to the degrading performance on the out-of-domain task when a model is fine-tuned on in-domain data. This setup is a popular topic in general machine learning research (Aljundi et al., 2019). Thompson et al. (2018) study domain adaptation by freezing subnetworks. Our work instead addresses forgetting in the data-balancing scenario for multi-domains. We use curriculum to generalize fine-tuning.

[1] We treat denoising as a domain in the paper, inspired by previous works that treat data noise using domain adaptation methods, e.g., (Junczys-Dowmunt, 2018).

[2] We refer to data regularization (using more data) and to transfer learning (fine-tuning) to exploit both data quantity and quality, the idea behind dynamic data selection. See Appendix C.

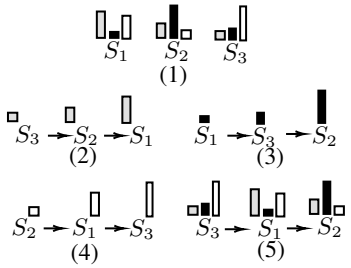

Figure 1: Data order in single-domain curricula and a potential multi-domain curriculum. (1) A toy training dataset of 3 examples. Each example has three scores, representing relevance to three domains, grey/dark/white domains, respectively. The higher the bar the more relevant. (2) Grey-domain order. (3) Dark-domain order. (4) White-domain order. (5) A potential multi-domain data order.

3 Curriculum Learning for NMT

We first introduce curriculum learning (CL) (Bengio et al., 2009), which serves as a formulation for SOTA single-domain dynamic data selection and which our method is built upon and generalizes. In CL, a curriculum, $\mathcal{C}$, is a sequence of training criteria over training steps. A training criterion, $Q_t(y|x)$, at step t is associated with a set of weights, $W_t(x, y)$,[3] over training sentence pairs (x, y) in a parallel dataset D, where y is the translation for x. $Q_t(y|x)$ is a re-weighting of the original training distribution $P(y|x)$:

$$Q_t(y|x) \propto W_t(x, y)P(y|x), \quad \forall (x, y) \in D \quad (1)$$

Hence, for T maximum training steps, $\mathcal{C}$ is a sequence:

$$\mathcal{C} = \langle Q_1, ..., Q_t, ..., Q_T \rangle \quad (2)$$

At step t, an online learner randomly samples a data batch from Q_t to fine-tune model m_{t-1} into m_t. Therefore, $\mathcal{C}$ corresponds to a sequence of models,

$$\langle m_1, ..., m_t, ..., M \rangle. \quad (3)$$

M is the final model that the entire curriculum has been optimizing towards. Intermediate models, m_t, serve as "stepping stones" to M, to transfer knowledge through them and regularize the training for generalization. A performance metric $\mathcal{P}(\mathcal{C})$ evaluates M on a development or test set, after training on $\mathcal{C}$.

[3] As a preview, in our paper, $W_t(x, y)$ uses uniform weights over selected examples and assigns zero weights for filtered examples, similar to a mask.

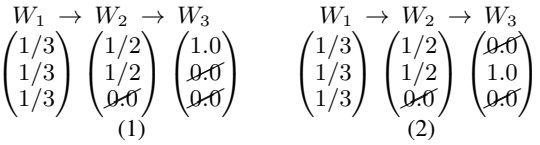

Table 2: Curriculum examples characterized by re-weighting, $W_t(x, y)$, over three steps, to stochastically order data to benefit a final domain. Strikethrough discards examples. (1) corresponds to data order Figure 1 (2). (2) corresponds to data order Figure 1 (5).

In NMT, CL is used to implement dynamic data selection. First, a scoring function (Section 4.3) is employed to measure the usefulness of an example to a domain and sort data. Then mini-batch sampling, e.g., (Kocmi and Bojar, 2017), is designed to realize the weighting W_t, to dynamically evolve the training criteria Q_t towards in-domain. Figure 1 (1)-(4) illustrates the basic idea of the curriculum we use. (1) shows three sentence pairs, S_1, S_2, S_3, each having three scores, respectively representing usefulness to three domains. A grey-domain training curriculum, for example, relies on the data order in (2), gradually discards least useful examples according to $W_t(x, y)$ (Eq. 1) in Table 2 (1): At step 1, the learner uniformly samples from all examples (W_1), producing model m_1. In step 2, the least-in-domain S_3 is discarded (strikethrough) by W_2 so we sample from subset $\{S_1, S_2\}$ uniformly to reach m_2. We repeat this until reaching the final model M. In this process, sampling is uniform in each step, but in-domain examples (e.g., S_1) are reused more over steps. Similarly, we can construct the dark-domain curriculum in Figure 1 (3) and the white-domain (4).

4 Our Approach: Learning a Multi-Domain Curriculum

4.1 General Idea

The challenges in multi-domain/-task data selection lie in addressing catastrophic forgetting and data balancing. In Figure 1, while curriculum (2) moves a model to the grey-domain direction, this direction may not necessarily be positively consistent with the dark domain (Figure 1 (3)), causing dropped dark-domain performance. Ideally, a training example that introduces the least forgetting across all domains would have gradients that move the model in a common direction towards all domains. While this may not be easily feasible by selecting a single example, we would like the intuition to work in a data batch on average. Therefore, our idea is to carefully introduce

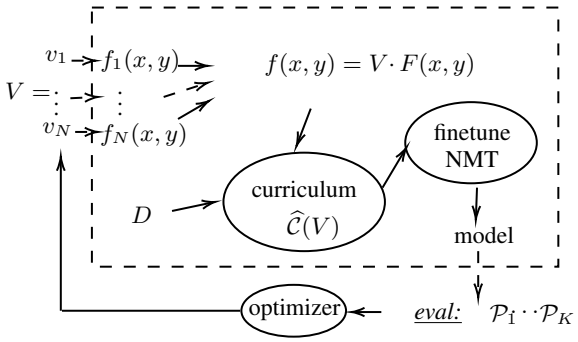

Figure 2: Learning a multi-domain curriculum.

per-example data-selection scores (called features) to measure "domain sharing", intelligently weight them to balance the domains of interest, and dynamically schedule examples to trade-off between regularization and domain adaptation.

A method to realize the above idea has the following *properties*:

1. Features of an example reflect its relevance to domains.
2. Feature weights are jointly learned/optimized based on end model performance.
3. Training is dynamic, by gradually focusing on multi-domain relevant and noise-reduced data batches.

Furthermore, a viable multi-domain curriculum meets the following *performance requirements*:

(i) It improves the baseline model across all domains.

(ii) It simultaneously reaches (or outperforms) the peak performance of individual single-domain curricula.

Above requires improvement over out-of-domain, too.

4.2 The Framework

Formally, for a sentence pair (x, y), let $f_n(x, y) \in \mathbb{R}$ be its n-th feature that specifies how (x, y) is useful to a domain. Suppose we are interested in K domains and each example has N features. For instance, each sentence pair of $S1, S2, S3$ in Figure 1 (1) has three features ($N = 3$), each for one domain ($K = 3$).[4] We represent (x, y)'s features using a feature vector $F(x, y) = [f_0(x, y), ..., f_{N-1}(x, y)]$. Given a weight vector $V = [v_0, ..., v_{N-1}]$ for all sentence pairs, we compute an aggregated score

$$f(x, y) = V \cdot F(x, y) \tag{4}$$

for each sentence pair and sort the entire data in increasing order. We then construct a curriculum $\widehat{\mathcal{C}}(V)$ to *fine-tune* a warmed-up model, evaluate its performance and propose a next weight vector. After several iterations/trials, the optimal weight vector V^* is the one with the best end performance:

$$V^* = \arg\max_V \mathcal{P}(\widehat{\mathcal{C}}(V)) \tag{5}$$

Figure 2 shows the framework. For the process to be practical and scalable, $\widehat{\mathcal{C}}$ fine-tunes a warmed-up model for a small number of steps. The learned V^* can then eventually be used for retraining a final model from scratch.

4.3 Instance-Level Features

We design the following types of features for each training example and instantiate them in Experiments (Section 5).

NMT domain features (q_Z) compute, for a pair (x, y), the cross-entropy difference between two NMT models:

$$q_Z(x, y) = \frac{\log P(y|x; \theta_Z) - \log P(y|x; \theta_{base})}{|y|} \tag{6}$$

$P(y|x; \theta_{base})$ is a baseline model with parameters θ_{base} trained on the background parallel corpus, $P(y|x; \theta_Z)$ is a Z-domain model with θ_Z by fine-tuning θ_{base} on a small, Z-domain parallel corpus $\widehat{D}_Z$ with trusted quality and $|y|$ is the length of y. q_Z discerns both noise and domain Z (Wang et al., 2019a). Each domain Z has its own $\widehat{D}_Z$.

Importantly, Grangier (2019) shows that, under the Taylor approximation (Abramowitz and Stegun, 1964), q_Z approximates the dot product between gradient, $g(x, y; \theta_{base})$, of training example (x, y) and gradient, $g(\widehat{D}_Z, \theta_{base})$, of seed data $\widehat{D}_Z$.[5] Thus an example with positive q_Z likely

[4] But N does not necessarily equal K because we can introduce multiple features for one domain or a single feature for multiple domains.

[5] That is, according to Grangier (2019):

$$\begin{aligned} q_Z(x, y) \times |y| &= \\ \log P(y|x; \theta_Z) - \log P(y|x; \theta_{base}) &\approx \\ \lambda\, g(x, y; \theta_{base})^\top g(\widehat{D}_Z, \theta_{base}) & \end{aligned} \tag{7}$$

when θ_{base} and θ_Z are close, which is the case for fine-tuning: $\theta_Z = \theta_{base} + \lambda\, g(\widehat{D}_Z, \theta_{base})$.

moves a model towards domain Z. For multiple domains, $Z_1, ..., Z_K$, selecting a batch of examples with q_{Z_k}'s all being positive would move a model towards a common direction shared across multiple domains, which alleviates forgetting.

The Z-domain feature $q_Z(x, y)$ can be easily generalized into a single *multi-domain feature*, $q_{\mathcal{Z}}$, for a set of domains $\mathcal{Z}$:

$$q_{\mathcal{Z}}(x, y) = \frac{\log P(y|x; \theta_{\mathcal{Z}}) - \log P(y|x; \theta_{base})}{|y|} \quad (8)$$

by simply concatenating all the seed parallel corpus $\widehat{D}_Z$ from the constituent domains into $\widehat{D}_{\mathcal{Z}}$ and use it to fine-tune the baseline θ_{base} into $\theta_{\mathcal{Z}}$. A benefit of $q_{\mathcal{Z}}$ is scalability: using a single feature value to approximate (x, y)'s gradient consistency with the multiple domains at once. Simple concatenation means, however, domain balancing is not optimized as in Eq. 5.

NLM domain features (d_Z) (Moore and Lewis, 2010; van der Wees et al., 2017) compute Z-domain relevance of sentence x with neural language models (NLM), like q_Z:

$$d_Z(x) = \frac{\log P(x; \vartheta_Z) - \log P(x; \vartheta_{base})}{|x|} \quad (9)$$

where $P(x; \vartheta_{base})$ is an NLM with parameters ϑ_{base} trained on the x half of the background parallel data, and $P(x; \vartheta_Z)$ is obtained by fine-tuning $P(x; \vartheta_{base})$ on Z-domain monolingual data. Although d_Z may not necessarily reflect the translation gradient of an example under an NMT model, it effectively assesses the Z-domain relevance and, furthermore, allows us to include additional larger amounts of in-domain monolingual data. We do not use its bilingual version (Axelrod et al., 2011), but choose to consider only the source side, for simplicity.

Cross-lingual embedding similarity feature (emb) computes the cosine similarity of a sentence pair in a cross-lingual embedding space. The embedding model is trained to produce similar representations exclusively for true bilingual sentence pairs, following Yang et al. (2019).

BERT quality feature (BERT) represents quality scores from a fine-tuned multilingual BERT model (Devlin et al., 2018). We fine-tune a pre-trained BERT model[6] on a supervised dataset with positive and negative translation pairs.

[6]We use the public cased 12 layers multilingual model: `multi_cased_L-12_H-768_A-12`

Algorithm 1: Bayesian optimization

```
H = ∅; # Trial history.
σ_0 = GP; # Initialize surrogate model.
α = EI; # Initialize acquisition function.
i = 1;
while i ≤ T do
    V_i = arg max_V α(V; σ_{i−1}, H); # Predict weights
          vector V_i by maximizing acquisition function.
    p = P(Ĉ(V_i)) by fine-tuning NMT on Ĉ(V_i);
    H = H ∪ {(V_i, p)}; # Update trial history.
    Estimate σ_i with H;
    i = i + 1;
end while
return (V*, p*) (∈ H) w/ the best performance p*.
```

These features compensate each other by capturing the information in a sentence pair from different aspects: NLM features capture domain. NMT features additionally discern noise. BERT and emb are introduced for denoising, by transfering the strength of the data they are trained on. All these features are from previous research and here we integrate them to solve a generalized problem.

4.4 Performance Metric $\mathcal{P}$

Eq. 5 evaluates the end performance $\mathcal{P}(\widehat{\mathcal{C}}(V))$ of a multi-domain curriculum candidate. We simply combine the validation sets from multi-domains into a single validation set to report the perplexity of the last model checkpoint, after training the model on $\widehat{\mathcal{C}}(V)$. The best multi-domain curriculum minimizes model's perplexity (or maximizes its negative per Eq. 5) on the mixed validation set. We experiment with different mixing ratios.

4.5 Curriculum Optimization

We solve Eq. 5 with Bayesian Optimization (BayesOpt) (Shahriari et al., 2016) as the optimizer in Figure 2. BayesOpt is derivative-free and can optimize expensive black-box functions, with no assumption of the form of $\mathcal{P}$. It has recently become popular for training expensive machine-learning models in the "AutoML" paradigm. It consists of a surrogate model for approximating $\mathcal{P}(\widehat{\mathcal{C}}(V))$ and an acquisition function for deciding the next sample to evaluate. The surrogate model evaluates $\widehat{\mathcal{C}}(V)$ without running the actual NMT training, by the Gaussian process (GP) priors over functions that express assumptions about $\mathcal{P}$. The acquisition function depends on previous trials, as well as the GP hyper-parameters. The Expected Improvement (EI) criterion (Srinivas et al., 2010) is usually used as acquisition function. Algo-

rithm 1 depicts how BayesOpt works in our setup. We use Vizier (Golovin et al., 2017) for Batched Gaussian Process Bandit, but open-source implementations of BayesOpt are easily available.[7].

4.6 Curriculum Construction

We pre-compute all features for each sentence pair (x, y) in training data and turn its features into a single score $f(x, y)$ by Eq. 4, given a weight vector. We then construct a curriculum by instantiating its re-weighting $W_t(x, y)$ (Eq. 1). To that end, we define a Boolean, dynamic data selection function $\chi^f_\rho(x, y; t)$ to check, at step t, if $(x, y) \in D$ belongs to the top $\rho(t)$-ratio examples in training data D sorted in increasing order of $f(x, y)$, $(0 < \rho \leq 1)$. So χ^f_ρ is a mask. Suppose $n(t)$ examples are selected by $\chi^f_\rho(x, y; t)$, the re-weighting will then be

$$W_t(x, y) = 1/n(t) \times \chi^f_\rho(x, y; t). \quad (10)$$

Filtered examples have zero weights and selected ones are uniformly weighted. We set $\rho(t) = (1/2)^{t/H}$ to decay/tighten over time[8], controlled by the hyper-parameter H. During training, $\chi^f_\rho(x, y; t)$ progressively selects higher $f(x, y)$-scoring examples. In implementation, we integrate $\chi^f_\rho(x, y; t)$ in the data feeder to pass only selected examples to the downstream model trainer; we also normalize $f(x, y)$ offline to directly compare to $\rho(t)$ online to decide filtering. As an example, the $W_t(x, y)$ for the multi-domain curriculum order in Figure 1 (5) can look like Table 2 (2).

5 Experiments

5.1 Setup

Data and domains. We experiment with two English→French training datasets: the noisy ParaCrawl data[9] (290 million sentence pairs) and the WMT14 training data (38 million pairs). We use SentencePiece model (Kudo, 2018) for subword segmentation with a source-target shared vocabulary of 32,000 subword units. We evaluate our method with three "domains": two specific domains, news and TED subtitles, and out-of-domain. News domain uses the WMT14 news testset (**N14**) for testing, and WMT12-13 for validation in early stopping (Prechelt, 1997). The TED domain uses the IWSLT15 testset (**T15**) for testing, and the IWSLT14 testset for validation. Out-of-domain performance is measured by two additional testsets, patent testset (**PA**) (2000 sentences)[10] and WMT15 news discussion testset (**D15**). We report **SacreBLEU**[11] (Post, 2018).

Features. NMT features use the parallel data to train the baseline NMT models. The new-domain-discerning NMT feature q_N uses WMT10-11 (5500 pairs) as in-domain data $\widehat{D}_N$. The TED NMT feature q_T uses the TED subtitle training data (22k pairs) as in-domain data $\widehat{D}_T$. NLM features use the English half of parallel data to train the baseline NLMs. The news-domain-discerning NLM feature d_N uses the 28 million English sentences from WMT14. The TED subtitle NLM feature d_T uses the English side of IWSLT15 in-domain parallel training data. The training of the cross-lingual embedding model follows Yang et al. (2019) with a 3-layer transformer (Vaswani et al., 2017) (more details in Appendix A). For the BERT feature, we sample positive pairs from the same data to train the cross-lingual embedding model. The negatives are generated using the cross-lingual embedding model, via 10-nearest neighbor retrieval in the embedding space, excluding the true translation. We pick the nearest neighbor to form a hard negative pair with the English sentence, and a random neighbor to form another negative pair. We sample 600k positive pairs and produce 1.8M pairs in total.

Model. We use LSTM NMT (Wu et al., 2016) as our models, but with the Adam optimizer (Kingma and Ba, 2015). The batch size is 10k averaged over 8 length-buckets (with synchronous training). NLM/NMT features uses 512 dimensions by 3 layers–NLM shares the same architecture as NMT by using dummy source sentences (Sennrich et al., 2016). The final models are of 1024 dimensions by 8 layers, trained for $55k$ max steps. Training on WMT data uses a dropout probability of 0.2. Transformer results are in Appendix B.

Curriculum optimization. In Eq. 5 (Section 4.5), we launch 30 trials (candidate curricula). BayesOpt spends 25 trials in exploration

[7] E.g., https://github.com/tobegit3hub/advisor

[8] When the training data is small, we can, in practice, let a model warm up before applying the schedule.

[9] https://paracrawl.eu

[10] Randomly sampled from www.epo.org

[11] Signature: BLEU+case.mixed+numrefs.1+smooth.exp+tok.13a+version.1.4.2

Curriculum	N14	T15	PA	D15	Avg
P1: $\mathcal{B}$	33.4	35.7	29.8	30.4	32.3
P2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**37.0**	**38.1**	**48.3**	**35.7**	**39.8**
W1: $\mathcal{B}$	$38.0_{39.2}$	37.9	45.6	34.5	39.0
(Wu et al., 2016)	*39.2*	–	–	–	–
W2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**39.3**	**38.8**	**46.1**	**36.1**	**40.1**

Table 3: English→French multi-domain curriculum improves no-curriculum baseline ($\mathcal{B}$) over all testsets. Avg: averaged score per row, for ease of reading. P: ParaCrawl data. W: WMT14 training data. BLEUs in italics are tokenized BLEU. Other scores are de-tokenized SacreBLEU.

and the last 5 in exploitation. Each trial trains for $2k$ steps[12] by fine-tuning a warmed-up model with the candidate curriculum. The curriculum decays ($\rho(t)$) from 100% and plateaus at 20% at step 2k. We simply and heuristically set a range of $[0.0, 1.0]$ for all feature weights. We don't normalize feature values when weighting them.

5.2 Results

We evaluate if the multi-domain curriculum meets requirements (i) and (ii) in Section 4.1.

5.2.1 Compared to no curriculum

We compare:

- $\mathcal{B}$: baseline that does not use curriculum learning.
- $\widehat{\mathcal{C}}_{\text{6-feats}}$: multi-domain curriculum with 6 features, d_N, d_T, q_N, q_T, BERT, emb, weights learned by BayesOpt.

Table 3 shows $\widehat{\mathcal{C}}_{\text{6-feats}}$ improves $\mathcal{B}$ on all testsets, especially on noisy ParaCrawl–requirement (i) is met. It is important to note that our WMT baseline (W1) matches Wu et al. (2016) on N14, as shown by re-computed tokenized BLEU (italics).

5.2.2 Compared to single-domain curricula

We examine the following individual curricula, by training NMT models with each, respectively:

- $\mathcal{C}_{d_N}$, uses news NLM feature d_N (Eq. 9).
- $\mathcal{C}_{d_T}$, uses TED subtitle NLM feature d_T.
- $\mathcal{C}_{q_N}$, uses news NMT feature q_N (Eq. 6).
- $\mathcal{C}_{q_T}$, uses TED NMT feature q_T.
- $\mathcal{C}_{\text{BERT}}$, uses BERT quality feature.
- $\mathcal{C}_{\text{emb}}$, uses cross-lingual embedding feature.

[12] 2k is empirically chosen to be practical. We use a number of fine-tuning trials in Eq. 5. NMT training is expensive so we don't want a trial to tune for many steps. NMT is very adaptive on domain data, so each trial does not need many steps. We find no significant difference among 1k, 2k, 6k.

Curriculum	N14	T15	PA	D15	Avg
P1: $\mathcal{B}$	33.4	35.7	29.8	30.4	32.3
P3: $\mathcal{C}_{d_N}$	34.7	36.2	32.6	32.6	34.0
P4: $\mathcal{C}_{d_T}$	34.8	36.3	30.1	32.4	33.4
P5.1: $\mathcal{C}_{\text{BERT}}$	36.8	37.3	47.9	35.0	39.3
P5.2: $\mathcal{C}_{\text{emb}}$	36.9	37.7	46.0	35.2	39.0
P6: $\mathcal{C}_{q_N}$	36.8	37.1	47.7	34.9	39.1
P7: $\mathcal{C}_{q_T}$	35.6	38.3	46.6	34.9	38.9
P2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**37.0**	38.1	**48.3**	**35.7**	**39.8**
P2 – P*	+0.1	-0.2	+0.4	+0.5	+0.2
W1: $\mathcal{B}$	38.0	37.9	45.6	34.5	39.0
W3: $\mathcal{C}_{d_N}$	38.3	38.1	39.1	35.1	37.7
W4: $\mathcal{C}_{d_T}$	38.1	38.4	43.0	36.1	38.9
W5.1: $\mathcal{C}_{\text{BERT}}$	38.5	37.8	45.9	35.9	39.5
W5.2: $\mathcal{C}_{\text{emb}}$	38.5	37.8	45.8	35.9	39.5
W6: $\mathcal{C}_{q_N}$	37.8	38.0	45.9	35.3	39.3
W7: $\mathcal{C}_{q_T}$	38.5	38.8	45.0	36.1	39.6
W2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**39.3**	**38.8**	**46.1**	**36.1**	**40.1**
W2 – W*	+0.8	0.0	+0.2	0.0	+0.3

Table 4: English→French multi-domain curriculum (P2, W2) vs. single-domain curricula (P3-7, W3-7). Frame boxes mark best per-testset BLEU (W*, P*) over all single-domain curricula. Bold color denotes multi-domain curriculum has best BLEU (W2-W* $\geq$ 0).

In Table 4, frame boxes mark the best BLEUs (P* or W*) per column, across P3-P7 or W3-W7. The last column shows averaged BLEU over all testsets. Bold font indicates $\mathcal{C}_{\text{6-feats}}$ matches or improves W*. As shown, $\mathcal{C}_{\text{6-feats}}$ matches or slightly outperforms the per-domain curricula across testsets. Therefore, $\widehat{\mathcal{C}}_{\text{6-feats}}$ meets requirement (ii).

5.3 Ablation Studies

5.3.1 Features

Strengths and weaknesses of a feature. Table 4 also reveals the relative strengths and weaknesses of each type of features. The peak BLEU (in a frame box) on each testset is achieved by one of $\mathcal{C}_{\text{BERT/emb}}$, $\mathcal{C}_{q_N}$ and $\mathcal{C}_{q_T}$, less by NLM features d_N, d_T. This contrast seems bigger on the noisy ParaCrawl, but the NLM features do bring gains over $\mathcal{B}$. Overall, $\mathcal{C}_{\text{BERT/emb}}$ (P5, W5) perform well, attributed to their denoising power, but lose to the NMT features (P7, W7) on T15, due to lack of explicit capturing of domain. The NMT features seem to subtly compensate in domains, and the domain features in denoising, but working with other features improves the model.

BERT and emb features. Both BERT and emb use knowledge external to the experiment setup. For a fair comparison to baselines and a better understanding of them, we drop them by building

Curriculum	N14	T15	PA	D15	Avg
P2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**37.0**	**38.1**	**48.3**	**35.7**	**39.8**
P8: $\widehat{\mathcal{C}}_{\text{4-feats}}$	36.6	**38.1**	46.7	35.5	39.2
W2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**39.3**	38.8	46.1	**36.1**	**40.1**
W8: $\widehat{\mathcal{C}}_{\text{4-feats}}$	38.9	**38.9**	**46.5**	**36.1**	**40.1**

Table 5: BERT and emb features positively contribute to $\widehat{\mathcal{C}}_{\text{6-feats}}$ on ParaCrawl (P).

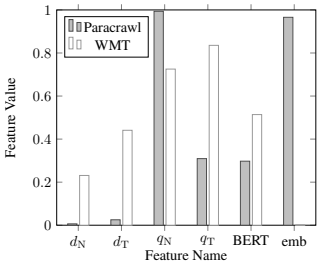

Figure 3: BayesOpt learns to weight features adaptively on ParaCrawl and WMT, respectively.

- $\widehat{\mathcal{C}}_{\text{4-feats}}$, multi-domain curriculum that excludes BERT and emb and uses 4 features.

Table 5 shows BERT and emb features in $\widehat{\mathcal{C}}_{\text{6-feats}}$ improve $\widehat{\mathcal{C}}_{\text{4-feats}}$ with ParaCrawl, adding to the intuition that they have a denoising effect.

Learned feature weights. Figure 3 shows BayesOpt learns to weight features adaptively in $\widehat{\mathcal{C}}_{\text{6-feats}}$ on ParaCrawl (grey) and WMT (white), respectively. ParaCrawl is very noisy thus noise non-discerning features d_N and d_T do not have a chance to help, but their weights become stronger on the cleaner WMT training data. It is surprising that BERT feature is still useful to the WMT training. We hypothesize this may suggest BERT feature have additional strength to just denoising, or that data noise could be subtle and exist in cleaner data.

5.3.2 BayesOpt vs. random search

We compare BayesOpt (BO) and Random Search (RS) (Bergstra and Bengio, 2012) to solve Eq. 5, as well as uniform weighting (Uniform). In Table 6, all improve baselines, especially on ParaCrawl (P). RS does surprisingly well on ParaCrawl, but BayesOpt appears better overall.[13]

5.3.3 Mixing validation sets

Eq. 5 evaluates $\mathcal{P}$ using the concatenated validation set (Section 4.4). Table 7 shows that the news-vs-TED mixing ratios can affect the per-domain BLEUs. For example, on ParaCrawl, when news sentences are absent from the validation set, N14 drops by 0.7 BLEU (P8 vs. P13). We use the four feats as in $\widehat{\mathcal{C}}_{\text{4-feats}}$ in this examination.

[13] RS uses 30 trials, as BO (Section 5.1), so the results show their comparison given the same number of trials.

Curriculum	N14	T15	PA	D15	Avg
P1 : $\mathcal{B}$	33.4	35.7	29.8	30.4	32.3
P2 : $\widehat{\mathcal{C}}_{\text{6-feats}}$ (BO)	**37.0**	38.1	**48.3**	**35.7**	**39.8**
P9 : $\widehat{\mathcal{C}}_{\text{6-feats}}$ (RS)	36.7	**38.4**	48.0	35.5	39.7
P10: $\widehat{\mathcal{C}}_{\text{6-feats}}$ (Uniform)	35.4	36.9	**48.3**	34.1	38.7
W1 : $\mathcal{B}$	38.0	37.9	45.6	34.5	39.0
W2 : $\widehat{\mathcal{C}}_{\text{6-feats}}$ (BO)	**39.3**	38.8	**46.1**	36.1	**40.1**
W9 : $\widehat{\mathcal{C}}_{\text{6-feats}}$ (RS)	39.0	38.2	43.7	**36.4**	39.3
W10: $\widehat{\mathcal{C}}_{\text{6-feats}}$ (Uniform)	38.8	**39.1**	43.0	36.0	39.2

Table 6: On average, BayesOpt (BO) performs better than Random Search (RS) and uniform weighting (Uniform), for learning feature weights of a multi-domain curriculum.

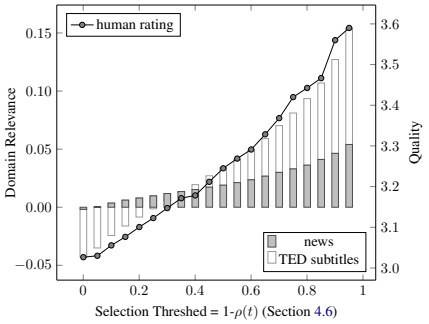

Figure 4: The multi-domain curriculum dynamically balances multi-domain-relevant and noise-reduced data, as validated by human ratings.

5.3.4 Dynamic data balancing

We simulate dynamic data selection with a random sample of 2000 pairs from the WMT data and annotate each pair by human raters with 0 (nonsense) - 4 (perfect) quality scale (following Wang et al. (2018b)). We sort the pairs by $f(x, y)$ (Eq. 4). A threshold selects a subset of pairs, for which we average the respective NMT feature values as the domain relevance. Figure 4 shows that the multi-domain curriculum ($\widehat{\mathcal{C}}_{\text{6-feats}}$) learns to dynamically increase quality and multi-domain relevance. Therefore, our idea (Section 4.1) works as intended. Furthermore, training seems to gradually increase quality or domain in different speeds, determined by Eq. 5.

5.3.5 Weighting loss vs. curriculum

With the learned weights, we compute a weight for each example to sort data to form a curriculum. Alternatively, we could weight the cross-entropy loss for that sentence during training (Wang et al., 2017; Chen et al., 2017). Table 8 shows that curriculum yields improvements over weighing per-

	Mixing Ratio	N14	T15	PA	D15	Avg
P11:	1.0:0.0	36.3	37.8	47.3	35.3	39.2
P12:	0.8:0.2	36.4	**38.2**	**47.7**	35.4	**39.4**
P8:	0.5:0.5	**36.6**	38.1	46.7	**35.5**	39.2
P13:	0.0:1.0	35.9	38.1	47.0	35.2	39.1
W11:	1.0:0.0	39.1	38.6	46.4	36.0	40.0
W12:	0.8:0.2	39.0	38.7	46.3	35.7	39.9
W8:	0.5:0.5	38.9	**38.9**	**46.5**	**36.1**	**40.1**
W13:	0.0:1.0	**39.1**	38.6	46.4	36.0	40.0

Table 7: Guiding multi-domain curriculum learning by mixing validation sets. Experiments use 4 features as in $\widehat{\mathcal{C}}_{\text{4-feats}}$.

Model	N14	T15	PA	D15	Avg
P8: Curriculum	**36.6**	**38.1**	**46.7**	**35.5**	**39.2**
P14: Weight Loss	35.3	37.8	39.3	32.6	36.3
W8: Curriculum	**38.9**	**38.9**	**46.5**	**36.1**	**40.1**
W14: Weight Loss	38.6	37.6	45.7	35.3	39.3

Table 8: Forming a curriculum with learned weights performs better than weighting instance loss in training. Experiments use 4 features (as in $\widehat{\mathcal{C}}_{\text{4-feats}}$).

sentence loss, in particular on noisy training data, confirming previous findings (van der Wees et al., 2017).

5.3.6 In-domain fine-tuning

$\mathcal{C}_{q_N}$ and $\mathcal{C}_{q_T}$ each use a small in-domain parallel dataset, but we can simply fine-tune the final models on either dataset (+N, +T) or their concatenation (+N+T). Table 9 shows that $\widehat{\mathcal{C}}_{\text{6-feats}}$ can be further improved by in-domain fine-tuning[14] and that both $\widehat{\mathcal{C}}_{\text{6-feats}}$ and its fine-tuning still improve the fine-tuned baselines, in particular on ParaCrawl.

5.4 Discussion: Feature Dependency

One potential issue with using multiple *per-domain* features ($q_Z(x, y)$'s in Eq. 6) is scores are not shared across domains and linear weighting may not capture feature dependency. For example, we need two NMT features if there are two domains. We replace the two NMT features, q_N and q_T, in $\widehat{\mathcal{C}}_{\text{4-feats}}$ with a single two-domain feature $q_{\mathcal{Z}=\{N,T\}}$ (Eq. 8), but with the two corresponding NLM features unchanged (so the new experiment has 3 features). Table 10 shows multi-domain feature contributes slightly better than linear combination of per-domain features (P19 vs. P8). The per-domain features, however, have the advantage of efficient feature weighting. In case of many features, learning to compress them seems to be an interesting future investigation.

[14] We fine-tune with SGD for 20k steps, with batch size 16, learning rate 0.0001.

Model	N14	T15	PA	D15	Avg
P15: $\mathcal{B}$+N	35.8	37.1	41.2	32.8	36.7
P16: $\mathcal{B}$+T	35.8	38.7	45.4	34.6	38.6
P17: $\mathcal{B}$+N+T	35.9	38.7	44.8	34.4	38.4
P2 : $\widehat{\mathcal{C}}_{\text{6-feats}}$ (BO)	37.0	38.1	48.3	35.7	39.8
P18: $\widehat{\mathcal{C}}_{\text{6-feats}}$ +N+T	**38.1**	**39.7**	**48.6**	**36.6**	**40.8**
W15: $\mathcal{B}$+N	38.7	37.4	**46.4**	34.6	39.3
W16: $\mathcal{B}$+T	36.8	38.9	44.8	36.5	39.3
W17: $\mathcal{B}$+N+T	38.6	39.1	46.1	35.8	39.9
W2 : $\widehat{\mathcal{C}}_{\text{6-feats}}$ (BO)	**39.3**	38.8	46.1	36.1	40.1
W18: $\widehat{\mathcal{C}}_{\text{6-feats}}$ +N+T	**39.3**	**39.8**	46.0	**36.6**	**40.4**

Table 9: The multi-domain curricula still bring improvements, even after models are fine-tuned on in-domain parallel data. +N: fine-tune on news parallel data $\widehat{D}_N$ (Section 5.1); +T: fine-tune on TED parallel data $\widehat{D}_T$; +N+T on concatenation.

Model	N14	T15	PA	D15	Avg
P8: per-dom.	**36.6**	38.1	46.7	35.5	39.2
P19: multi-dom.	**36.6**	**38.6**	**46.8**	**35.9**	**39.5**

Table 10: Multi-domain/task feature (Eq. 8) seems to contribute slightly better than linear combination of multiple per-domain features (Eq. 6).

6 Conclusion

Existing curriculum learning research in NMT focuses on a single domain. We present a multi-domain curriculum learning method. We carefully introduce instance-level features and learn a training curriculum to gradually concentrate on multi-domain relevant and noise-reduced data batches. End-to-end experiments and ablation studies on large datasets at different noise levels show that the multi-domain curriculum simultaneously reaches or outperforms the individual performance and brings solid gains over no-curriculum training, on in-domain and out-of-domain testsets.

Acknowledgments

The authors would like to thank David Grangier for Eq. 7 and derivation, the three anonymous reviewers for their insightful reviews, Yuan Cao for his technical suggestions, Jason Smith, Markus Freitag, Pidong Wang and Reid Pryzant for comments on an earlier draft, Quoc V. Le for suggestions in a related thread.

References

Milton Abramowitz and Irene A. Stegun. 1964. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, ninth dover printing, tenth gpo printing edition. Dover, New York.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. 2019. Gradient based sample selection for online continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 11816–11825. Curran Associates, Inc.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26 th International Conference on Machine Learning*, pages 86–96, Montreal, Canada.

James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305.

Denny Britz, Quoc Le, and Reid Pryzant. 2017. Effective domain mixing for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 118–126. Association for Computational Linguistics.

Boxing Chen, Colin Cherry, George Foster, and Samuel Larkin. 2017. Cost weighting for neural machine translation domain adaptation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 40–46, Vancouver. Association for Computational Linguistics.

Boxing Chen and Fei Huang. 2016. Semi-supervised convolutional networks for translation adaptation with tiny amount of in-domain data. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, pages 314–323.

Boxing Chen, Roland Kuhn, George Foster, Colin Cherry, and Fei Huang. 2016. Bilingual methods for adaptive training data selection for machine translation. In *AMTA*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Kevin Duh. 2018. Multi-objective hyperparameter search for fast and accurate neural machine translation - progress report. Technical report, Johns Hopkins University.

M. Amin Farajian, Marco Turchi, Matteo Negri, and Marcello Federico. 2017. Multi-domain neural machine translation through unsupervised adaptation. In *Proceedings of the Second Conference on Machine Translation*, pages 127–137, Copenhagen, Denmark. Association for Computational Linguistics.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *CoRR*, abs/1612.06897.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Elliot Karro, and D. Sculley, editors. 2017. *Google Vizier: A Service for Black-Box Optimization*.

IJ Goodfellow, M Mirza, X Da, Aaron Courville, and Yoshua Bengio. 2014. An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks. *TR arXiv:1312.6211v2*.

David Grangier. 2019. Distribution matched contrastive data selection. In *Google internal report*.

Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. *CoRR*, abs/1704.03003.

Marcin Junczys-Dowmunt. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 901–908, Belgium, Brussels. Association for Computational Linguistics.

Huda Khayrallah, Brian Thompson, Kevin Duh, and Philipp Koehn. 2018. Regularized training objective for continued training for domain adaptation in neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 36–44, Melbourne, Australia. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 379–386. INCOMA Ltd.

Philipp Koehn, Francisco Guzman, Vishrav Chaudhary, and Juan Pino. 2019. Findings of the wmt 2019 shared task on parallel corpus filtering for low-resource conditions. In *Proceedings of the Fourth Conference on Machine Translation*, Florence, Italy. Association for Computational Linguistics.

Philipp Koehn, Huda Khayrallah, Kenneth Heafield, and Mikel L. Forcada. 2018. Findings of the wmt 2018 shared task on parallel corpus filtering. In *Proceedings of the Third Conference on Machine Translation*, Belgium, Brussels. Association for Computational Linguistics.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75. Association for Computational Linguistics.

Gaurav Kumar, George Foster, Colin Cherry, and Maxim Krikun. 2019. Reinforcement learning based curriculum optimization for neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2054–2061, Minneapolis, Minnesota. Association for Computational Linguistics.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference*, pages 220–224.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabás Póczos, and Tom M. Mitchell. 2019. Competence-based curriculum learning for neural machine translation. *CoRR*, abs/1903.09848.

Matt Post. 2018. A call for clarity in reporting bleu scores. *Computing Research Repository*, arXiv:1804.08771v1. Version 2.

Lutz Prechelt. 1997. Automatic early stopping using cross validation: Quantifying the criteria. *Neural Networks*, 11:761–767.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. *CoRR*, abs/1707.05246.

Hassan Sajjad, Nadir Durrani, Fahim Dalvi, Yonatan Belinkov, and Stephan Vogel. 2017. Neural machine translation training in a multi-domain scenario. *arXiv preprint arXiv:1708.08712v2*.

Danielle Saunders, Felix Stahlberg, Adrià de Gispert, and Bill Byrne. 2019. Domain adaptive inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 222–228, Florence, Italy. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. 2016. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104:148–175.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. 2010. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 1015–1022, USA. Omnipress.

Sander Tars and Mark Fishel. 2018. Multi-domain neural machine translation. *CoRR*, abs/1805.02282.

Brian Thompson, Huda Khayrallah, Antonios Anastasopoulos, Arya D. McCarthy, Kevin Duh, Rebecca Marvin, Paul McNamee, Jeremy Gwinnup, Tim Anderson, and Philipp Koehn. 2018. Freezing subnetworks to analyze domain adaptation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 124–132, Brussels, Belgium. Association for Computational Linguistics.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Brian MacWhinney, and Chris Dyer. 2016. Learning the curriculum with bayesian optimization for task-specific word representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 130–139, Berlin, Germany. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488. Association for Computational Linguistics.

Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2018a. Dynamic sentence sampling for efficient training of neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 298–304, Melbourne, Australia. Association for Computational Linguistics.

Wei Wang, Isaac Caswell, and Ciprian Chelba. 2019a. Dynamically composing domain-data selection with clean-data selection by "co-curricular learning" for neural machine translation. In *Proceedings of the*

57th Conference of the Association for Computational Linguistics, pages 1282–1292, Florence, Italy. Association for Computational Linguistics.

Wei Wang, Taro Watanabe, Macduff Hughes, Tetsuji Nakagawa, and Ciprian Chelba. 2018b. Denoising neural machine translation training with trusted data and online data selection. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 133–143. Association for Computational Linguistics.

Xinyi Wang, Hieu Pham, Paul Michel, Antonios Anastasopoulos, Graham Neubig, and Jaime Carbonell. 2019b. Optimizing data usage via differentiable rewards. *arXiv preprint arXiv:1911.10088*.

Xinyi Wang, Yulia Tsvetkov, and Graham Neubig. 2020. Balancing training for multilingual neural machine translation. *arXiv preprint arXiv:2004.06748*.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine transaltion. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Yinfei Yang, Gustavo Hernández Ábrego, Steve Yuan, Mandy Guo, Qinlan Shen, Daniel Cer, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2019. Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax. *CoRR*, abs/1902.08564.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J. Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat. 2018. An empirical exploration of curriculum learning for neural machine translation. *CoRR*, abs/1811.00739.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. 2019. Curriculum learning for domain adaptation in neural machine translation. In *2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Appendices

A Cross-lingual Embedding Model Parameters

The sentence encoder has a shared 200k token multilingual vocabulary with 10k OOV buckets. For each token, we also extract character n-grams ($n = [3, 6]$) hashed to 200k buckets. Word token items and character n-gram items are mapped to 320 dim. character embeddings. Word and character n-gram representations are summed together to produce the final input token representation. The encoder is a 3-layer Transformer with hidden size of 512, filter size of 2048, and 8 attention heads. We train for 40M steps using an SGD optimizer with batch size K=100 and learning rate 0.003. During training, the word and character embeddings are scaled by a gradient multiplier of 25.

B Transformer-Big Results

We replicate experiments with the Transformer-Big architecture. Table 11 shows the Transformer-Big results that correspond to the RNN results in Table 3. These results show that the multi-domain curriculum meets the performance requirement (i) (Section 4.1) using the Transformer architecture. Table 12 shows the Transformer-Big results corresponding to RNN results in Table 4. They show that the proposed multi-domain curriculum meets the performance requirement (ii) using Transformer.

Curri.	N14	T15	PA	D15	Avg
P1: $\mathcal{B}$	34.1	36.3	34.2	32.3	34.2
P2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**39.6**	**40.2**	**50.6**	**37.7**	**42.0**
W1: $\mathcal{B}$	40.8	39.9	46.0	37.8	41.1
W2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**41.8**	**41.2**	**48.1**	**38.8**	**42.5**

Table 11: Transformer Big SacreBLEU: English → French multi-domain curriculum improves no-curriculum baseline ($\mathcal{B}$) over all testsets, using Transformer-Big. P: Paracrawl training data. W: WMT14 training data.

C An Explanation: Noisy Data Useful in Low-Resource Setup

With noisy, limited data (e.g., 100k pairs), we can train a model A on all data, or a model B on the filtered subset (e.g., 10k). We can also fine-tune A on the filtered data, to produce model C. C could be better than A due to use of higher-quality data or better than B due to use of more

Curri.	N14	T15	PA	D15	Avg
P1: $\mathcal{B}$	34.1	36.3	34.2	32.3	34.2
P3: $\mathcal{C}_{d_N}$	33.7	36.1	32.7	32.5	33.8
P4: $\mathcal{C}_{d_T}$	35.3	37.7	32.8	34.0	35.0
P5: $\mathcal{C}_{\text{BERT}}$	39.2	40.1	49.7	37.5	41.6
P6: $\mathcal{C}_{q_N}$	38.9	39.8	48.9	36.9	41.1
P7: $\mathcal{C}_{q_T}$	37.3	40.4	44.7	36.2	39.7
P2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**39.6**	40.2	**50.6**	**37.7**	**42.0**
P2 – P*	+0.4	-0.2	+0.9	+0.2	+0.3
W1: $\mathcal{B}$	40.8	39.9	46.0	37.8	41.1
W3: $\mathcal{C}_{d_N}$	40.9	39.2	44.4	37.6	40.5
W4: $\mathcal{C}_{d_T}$	39.8	39.6	43.3	37.3	40.0
W5: $\mathcal{C}_{\text{BERT}}$	40.5	39.2	45.7	38.3	40.9
W6: $\mathcal{C}_{q_N}$	41.1	40.0	47.6	38.0	41.7
W7: $\mathcal{C}_{q_T}$	41.1	41.4	47.7	38.5	42.2
W2: $\widehat{\mathcal{C}}_{\text{6-feats}}$	**41.8**	41.2	**48.1**	**38.8**	**42.5**
W2 – W*	+0.7	-0.2	+0.4	+0.3	+0.3

Table 12: Transformer Big SacreBLEU: English $\rightarrow$ French multi-domain curriculum (P2, W2) vs. single-domain curricula (P3-7, W3-7). BLEU scores over 4 testsets and their average. Frame boxes mark best per-testset BLEU (W*, P*) over all single-domain curricula. Bold color denotes multi-domain curriculum has best BLEU (W2-W* $\geq$ 0). P: ParaCrawl training data. W: WMT14 training data.

data (200k>10k). Therefore, by "noisy data can be helpful", we refer to data regularization (using more data) and to transfer learning (fine-tuning) to exploit both data quantity and quality, the idea behind dynamic data selection.